Optimal Nonlinear Neural Network Controllers for Aircraft

Joint University Program Meeting
October 10, 2001

Nilesh V. Kulkarni

Advisors

Prof. Minh Q. Phan
Dartmouth College, Hanover, NH

Prof. Robert F. Stengel
Princeton University, NJ

Presentation Outline:

- Research goals.
- Definition of the problem.
- Parametric optimization approach.
- Modified Approach.
- Neural Network implementation.
- Linear System Implementation.
- Nonlinear System Implementation.
- Conclusions

Research Goals

- To come up with a control approach that is:
  - Optimal or approaching optimality in the limit
  - Applicable to both linear and non-linear systems
  - Data-based (no need for an explicit analytical model of of the system)
  - Adaptive to account for slowly time-varying dynamics dynamics and operating conditions.
- Application to aircraft.

General Problem Statement:

- For the system dynamics:

$$\boldsymbol{x}(\boldsymbol{k}+1) = \boldsymbol{f}[\boldsymbol{x}(\boldsymbol{k}), \boldsymbol{u}(\boldsymbol{k}), \boldsymbol{p}(\boldsymbol{k}), \boldsymbol{k}]$$

$$\boldsymbol{y}(\boldsymbol{k}) = \boldsymbol{h}[\boldsymbol{x}(\boldsymbol{k}), \boldsymbol{u}(\boldsymbol{k})]$$

- Find a control law:

$$\boldsymbol{u}(\boldsymbol{k}) = \boldsymbol{u}[\boldsymbol{x}(\boldsymbol{k})]$$

- To maximize a performance index (minimize a cost function)

$$\boldsymbol{J} = \frac{1}{2}\sum_{\boldsymbol{i}=1}^{\infty}[\boldsymbol{x}(\boldsymbol{k}+\boldsymbol{i})^{T}\boldsymbol{Q}\boldsymbol{x}(\boldsymbol{k}+\boldsymbol{i}) + \boldsymbol{u}(\boldsymbol{k}+\boldsymbol{i}-1)^{T}\boldsymbol{R}\boldsymbol{u}(\boldsymbol{k}+\boldsymbol{i}-1)]$$

Approaches:

- Dynamic optimization approaches:
  - Calculus of variations approach.
    - Euler-Lagrange equations.
  - Dynamic programming.
    - Hamilton-Jacobi-Bellman equation.
    - Specialization to Adaptive critic designs.
- Static optimization approach:
  - Parametric Optimization.
  - Cost-to-go approach.

Direct Parametric Optimization Approach:

- Methodology:

$$x(k) \longrightarrow \boxed{u(x(k), G)} \longrightarrow \boxed{f(x(k), u(k))} \longrightarrow x(k+1)$$

Find the unknown coefficients, *'G'* , that minimize the cost-to-go function

$$V(k,G) = \frac{1}{2}\sum_{i=1}^{r}[x(k+i)^{T} Q x(k+i) + u(k+i-1)^{T} R u(k+i-1)]$$

- Disadvantages:
  - This approach reduces to solving a static optimization problem which is highly nonlinear even for linear systems
  - Easily gets stuck in spurious local minima even for the case of finding a linear optimal controller for a linear system
  - Chance of finding a workable optimal controller using such an approach in practice is very limited.

Illustrative Example:

For a simple linear time invariant system,

$$\boldsymbol{x}(\boldsymbol{k}+1) = \boldsymbol{A}\boldsymbol{x}(\boldsymbol{k}) + \boldsymbol{B}\boldsymbol{u}(\boldsymbol{k})$$

and

$$\boldsymbol{u}(\boldsymbol{k}) = \boldsymbol{G}\boldsymbol{x}(\boldsymbol{k})$$

we can write,

$$\boldsymbol{x}(\boldsymbol{k}+\boldsymbol{i}) = (\boldsymbol{A}+\boldsymbol{B}\boldsymbol{G})^{\boldsymbol{i}}\,\boldsymbol{x}(\boldsymbol{k})$$

And so,

$$\boldsymbol{V}(\boldsymbol{k},\boldsymbol{G}) = \frac{1}{2}\sum_{\boldsymbol{i}=1}^{\boldsymbol{r}}[\boldsymbol{x}(\boldsymbol{k}+\boldsymbol{i})^{\boldsymbol{T}}\boldsymbol{Q}\boldsymbol{x}(\boldsymbol{k}+\boldsymbol{i}) + \boldsymbol{u}(\boldsymbol{k}+\boldsymbol{i}-1)^{\boldsymbol{T}}\boldsymbol{R}\boldsymbol{u}(\boldsymbol{k}+\boldsymbol{i}-1)]$$

$$= \frac{1}{2}\boldsymbol{x}(\boldsymbol{k})^{\boldsymbol{T}}\sum_{\boldsymbol{i}=1}^{\boldsymbol{r}}[(\boldsymbol{A}+\boldsymbol{B}\boldsymbol{G})^{\boldsymbol{i}^{\boldsymbol{T}}}\boldsymbol{Q}(\boldsymbol{A}+\boldsymbol{B}\boldsymbol{G})^{\boldsymbol{i}} + (\boldsymbol{A}+\boldsymbol{B}\boldsymbol{G})^{\boldsymbol{i}-1^{\boldsymbol{T}}}\boldsymbol{G}^{\boldsymbol{T}}\boldsymbol{R}\boldsymbol{G}(\boldsymbol{A}+\boldsymbol{B}\boldsymbol{G})^{\boldsymbol{i}-1}]\boldsymbol{x}(\boldsymbol{k})$$

As seen the cost-to-go function expressed with a single parameter *'G'*, is a highly nonlinear function of the parameter and as seen from several test examples was found found to contain several minima.

Modified Approach:

- Reformulate the control law:

$$\boxed{\begin{aligned} &\boldsymbol{u}(\boldsymbol{k}) = \boldsymbol{u}_1[\boldsymbol{x}(\boldsymbol{k}), \boldsymbol{G}_1] \\ &\boldsymbol{u}(\boldsymbol{k}+1) = \boldsymbol{u}_2[\boldsymbol{x}(\boldsymbol{k}), \boldsymbol{G}_2] \\ &\dots \\ &\boldsymbol{u}(\boldsymbol{k}+\boldsymbol{r}-1) = \boldsymbol{u}_{\boldsymbol{r}}[\boldsymbol{x}(\boldsymbol{k}), \boldsymbol{G}_{\boldsymbol{r}}] \end{aligned}}$$

'r' represents the order of approximation of the cost-to-go function

- Set up the cost-to-go function in terms of the 'G's':

$$\boldsymbol{V}(\boldsymbol{k}, \boldsymbol{G}_1, .., \boldsymbol{G}_{\boldsymbol{r}}) = \frac{1}{2}\sum_{\boldsymbol{i}=1}^{\boldsymbol{r}}[\boldsymbol{x}(\boldsymbol{k}+\boldsymbol{i})^{\boldsymbol{T}}\boldsymbol{Q}\boldsymbol{x}(\boldsymbol{k}+\boldsymbol{i}) + \boldsymbol{u}(\boldsymbol{k}+\boldsymbol{i}-1)^{\boldsymbol{T}}\boldsymbol{R}\boldsymbol{u}(\boldsymbol{k}+\boldsymbol{i}-1)]$$

Modified Approach…

- Find the G's by imposing the stationarity conditions:

$$\frac{\partial \boldsymbol{V}}{\partial \boldsymbol{G}_1} = 0; \frac{\partial \boldsymbol{V}}{\partial \boldsymbol{G}_2} = 0; \ldots \frac{\partial \boldsymbol{V}}{\partial \boldsymbol{G}_{\boldsymbol{r}}} = 0;$$

and

$$\boldsymbol{G}_2 = \boldsymbol{G}_2(\boldsymbol{f}, \boldsymbol{G}_1, \boldsymbol{Q}, \boldsymbol{R})$$

$$\ldots$$

$$\boldsymbol{G}_{\boldsymbol{r}} = \boldsymbol{G}_{\boldsymbol{r}}(\boldsymbol{f}, \boldsymbol{G}_{\boldsymbol{r}-1}, \boldsymbol{Q}, \boldsymbol{R})$$

- Solving the second set of equations is not as easy and even less implementable in terms of a control architecture.
- *x(k)*, the present state of the system appears as a coefficient in the stationarity conditions.
- By solving the stationarity conditions for multiple *x(k)'s*, presents enough equations for solving for the unknown G's without solving the the second set of conditions.

Illustrative Example:

For a simple linear time invariant system,

$$\boldsymbol{x(k+1) = Ax(k) + Bu(k)}$$

we can write,

$$\boldsymbol{u(k) = G_1 x(k)}$$

$$\boldsymbol{u(k+1) = G_2 x(k)}$$

$$\boldsymbol{\ldots}$$

$$\boldsymbol{u(k+r-1) = G_r x(k)}$$

And so,

$$\boldsymbol{x(k+i) = (A^i + A^{i-1}BG_1 + \ldots + ABG_{i-1} + BG_i)x(k)}$$

$$\boldsymbol{V(k) = \frac{1}{2} x(k)^T [(A^r + \ldots + ABG_{r-1} + BG_r)^T Q(A^r + \ldots + ABG_{r-1} + BG_r)}$$

$$\boldsymbol{+ \ldots + (A+BG_1)^T Q(A+BG_1) + G_r{}^T RG_r + \ldots + G_1{}^T RG_1]x(k)}$$

As seen the cost-to-go function now expressed with the parameters *'G's'*, is a quadratic function of the parameter and therefore has a single minimum

Role of Neural Networks

- For a nonlinear system, the controller is typically nonlinear. nonlinear.
- Cost-to-go function is a nonlinear function.
- Being universal function approximators, Neural Networks present themselves as ideal tools for handling nonlinear systems in the proposed Cost-to-go design approach
- Neural networks present a straightforward approach for making the design adaptive even in the case of a nonlinear system.

Formulation of the Control Architecture: NN Cost-to-go function Approximator

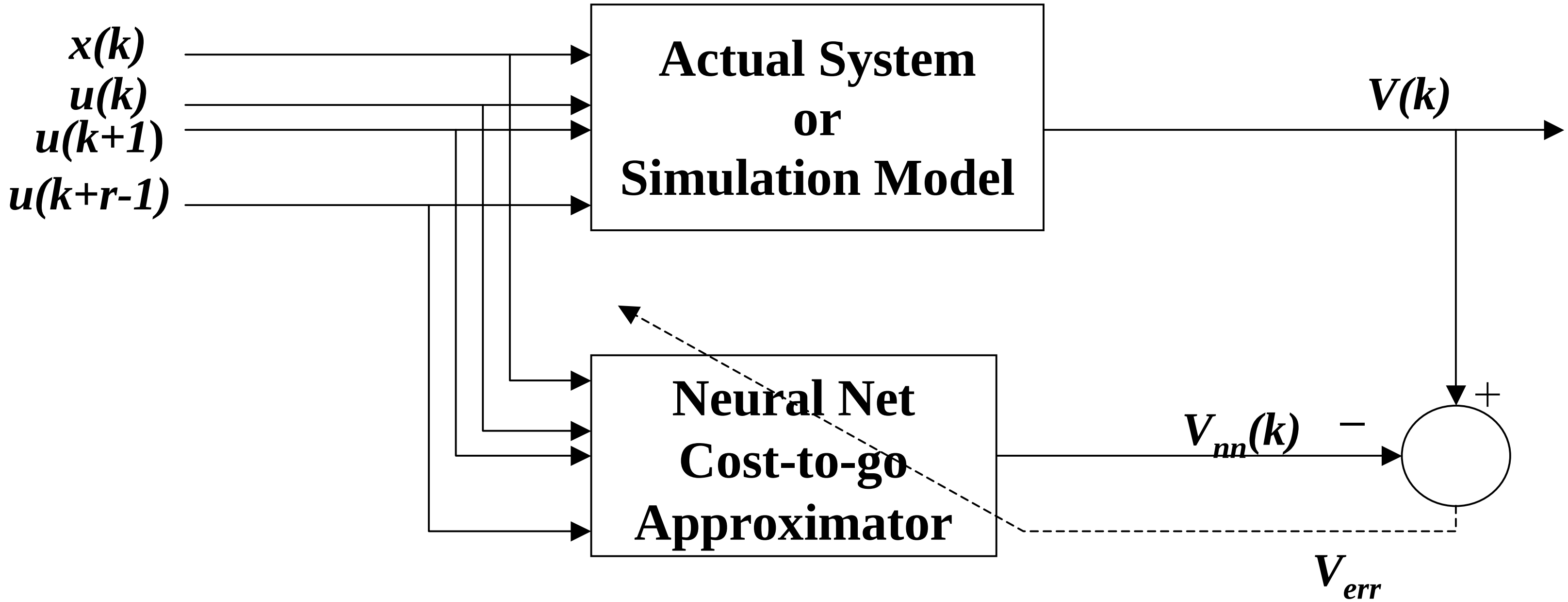
- Parameterize the cost-to-go function using a Neural Network (*CGA* (*CGA* Neural Network)
- Inputs to the *CGA* Network:

  x(k), u(k),…,u(k+r-1)
- Use the analytical model, or a computer simulation or the physical physical model to generate the future states.
- Use the *'r'* control values and the *'r'* future states to get the ideal ideal cost-to-go function estimate.

$$\boldsymbol{V}(\boldsymbol{k}) = \frac{1}{2}\sum_{\boldsymbol{i}=1}^{\boldsymbol{r}}[\boldsymbol{x}(\boldsymbol{k}+\boldsymbol{i})^T \boldsymbol{Q}\boldsymbol{x}(\boldsymbol{k}+\boldsymbol{i}) + \boldsymbol{u}(\boldsymbol{k}+\boldsymbol{i}-1)^T \boldsymbol{R}\boldsymbol{u}(\boldsymbol{k}+\boldsymbol{i}-1)]$$

- Use this to train the *CGA* Neural Network

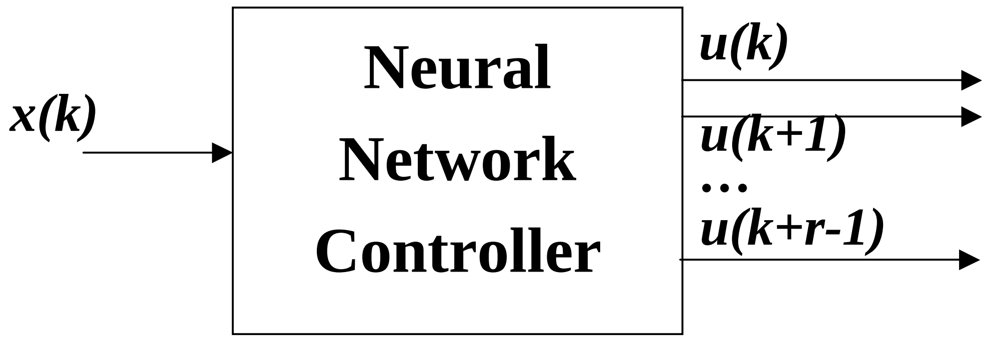
CGA Neural Network Training



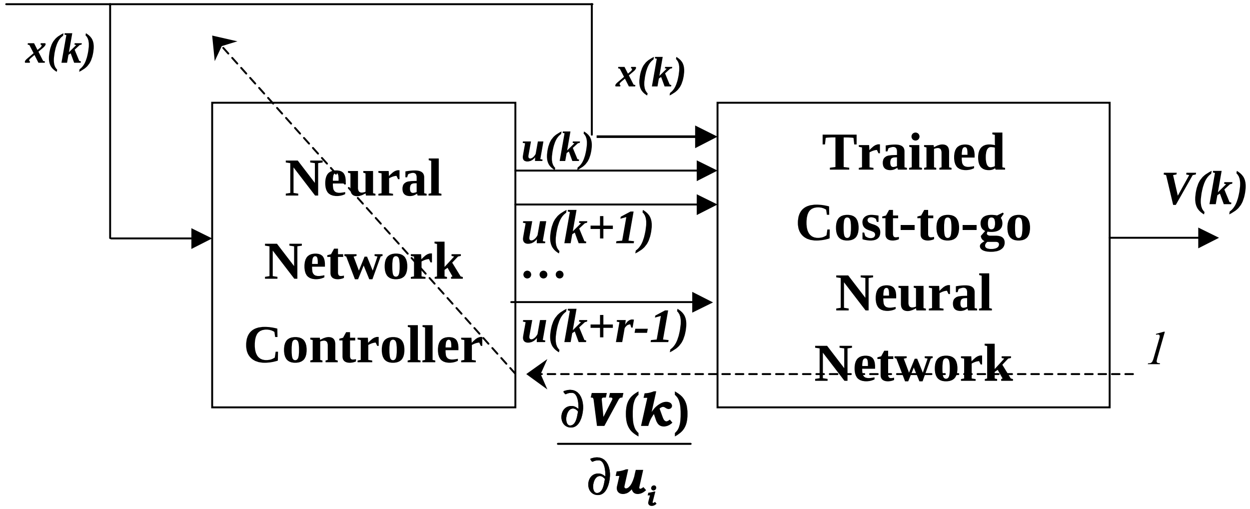
Neural Network Cost-to-go Approximator Training

Formulation of the Control Architecture: NN Controller

- Instead of a single controller structure (G), we need ‘r’ controller structures.
- The outputs of the ‘r’ controller structures, generate u(k) through u(k+r-1).
- Parameterize the ‘r’ controller structures using an effective Neural Network.



Neural Network Controller Training



- Gradient of *V(k)* with respect to the control inputs *u(k)* ,…., *u(k+r-1)* is calculated using back-propagation through the *'CGA'* Neural Network.
- These gradients can be further back-propagated through the Neural Network controller to get, $\frac{\partial V(k)}{\partial G_1}$ $\frac{\partial V(k)}{\partial G_r}$ through
- Neural Network controller is trained so that

$$\frac{\partial V(k)}{\partial G_i} \to 0, i = 1 \ldots r$$

Advantages of the formulation

- The modified parametric optimization simplifies the optimization problem.
- CGA Network training and the controller Network training is is decoupled.
- Implementation is system independent. So the basic architecture remains the same for linear or nonlinear systems. systems.
- Implementation is data-based. No explicit analytical model needed.
- Parameterization using Neural Networks makes the control architecture adaptive.
- Order of approximation *‘r’* in the definition of *V(k)* serves as as a tuning parameter.

Implementation for Linear Systems:

- Motivation:
  - Linear systems provide an easy way to see the details of the implementation of the cost-to-go design.
  - Provides a means for comparison of the results with existing solutions.

Optimal Control of Aircraft Lateral Dynamics

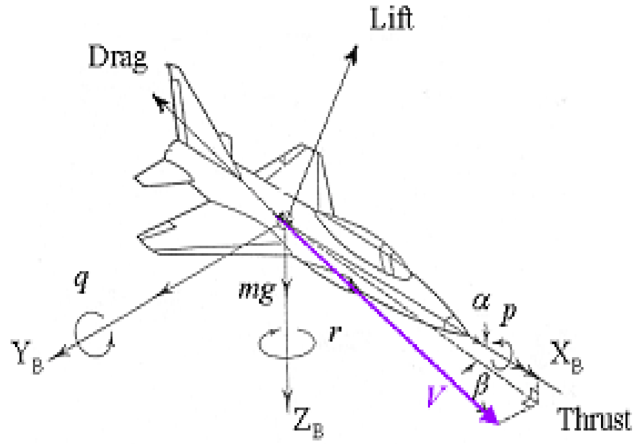
Airplane State Variables:

β - Side slip angle

p- Roll rate

r - Yaw rate

φ - Roll angle



Airplane Input Variables:

δ_r - Rudder Deflection

δ_a - Aileron Deflection

Phoenix Hobbico Hobbistar 60^{tm} model

$$\begin{bmatrix} \beta(k+1) \\ p(k+1) \\ r(k+1) \\ \varphi(k+1) \end{bmatrix} = \begin{bmatrix} 0.9811 & 0 & -0.0099 & 0.0036 \\ -0.0848 & 0.9665 & 0.0075 & -0.002 \\ 0.0690 & -0.0035 & 0.9992 & 0.0001 \\ -0.0004 & 0.0098 & 0 & 1 \end{bmatrix} \begin{bmatrix} \beta(k) \\ p(k) \\ r(k) \\ \varphi(k) \end{bmatrix} + \begin{bmatrix} 0 & 0.0529 \\ -0.0504 & 0.0637 \\ -0.0024 & -0.0179 \\ -0.0003 & 0.0003 \end{bmatrix} \begin{bmatrix} \delta_r \\ \delta_a \end{bmatrix}$$

$$\begin{bmatrix} \delta_r(k) \\ \delta_a(k) \end{bmatrix} = G\begin{bmatrix} \beta(k) & p(k) & r(k) & \varphi(k) \end{bmatrix}$$

$$V(k) = \sum_{i=k}^{k+r-1} \left(\begin{bmatrix} \beta(i+1) & p(i+1) & r(i+1) & \varphi(i+1) \end{bmatrix} \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 50 & 0 \\ 0 & 0 & 0 & 500 \end{bmatrix} \begin{bmatrix} \beta(i+1) \\ p(i+1) \\ r(i+1) \\ \varphi(i+1) \end{bmatrix} + \begin{bmatrix} \delta_a(i) & \delta_r(i) \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \delta_a(i) \\ \delta_r(i) \end{bmatrix} \right)$$

- **Optimal controller gains calculated using LQR optimal solution with perfect knowledge of the system:**

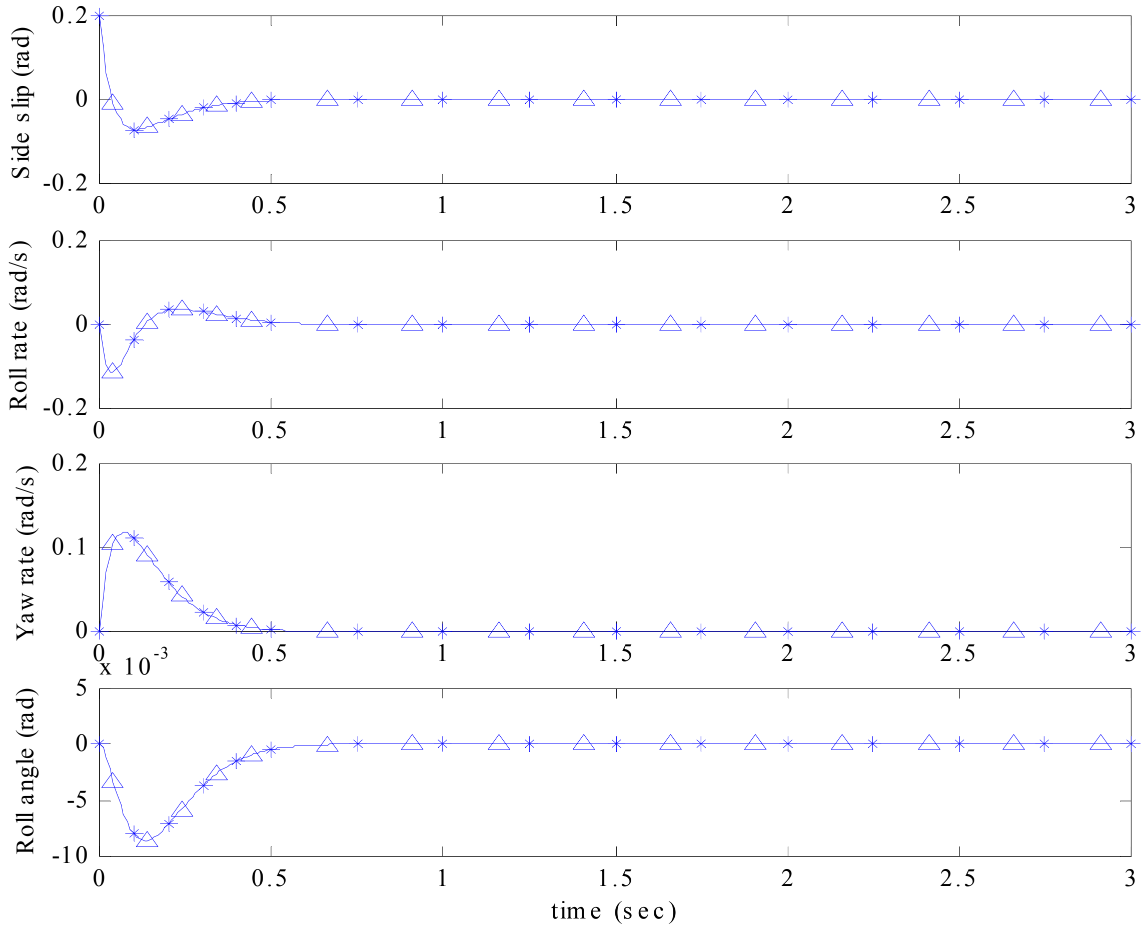
$$G=\begin{bmatrix} -4.3039 & 3.4120 & -1.1945 & 19.3901 \\ -7.0398 & -0.3318 & -5.0124 & -4.8498 \end{bmatrix}$$

- **Evaluated optimal cost** = *10.0925*

- **Gain obtained using the new data-based approach:**

Order of approximation (*'r'*)	*Evaluated cost ('J')*	Control gain
25	10.1598	$G=\begin{bmatrix} -3.7907 & 3.7868 & -0.7872 & 16.6319 \\ -6.5535 & -0.4424 & -4.5207 & -5.1355 \end{bmatrix}$
35	10.0935	$G=\begin{bmatrix} -4.2247 & 3.3616 & -1.1687 & 18.7413 \\ -6.9927 & -0.3411 & -4.9646 & -4.8351 \end{bmatrix}$
50	10.0925	$G=\begin{bmatrix} -4.3002 & 3.4077 & -1.1967 & 19.3265 \\ -7.0389 & -0.3313 & -5.0102 & -4.8365 \end{bmatrix}$

Comparison of the state trajectories using the cost-to-go design and the LQR design

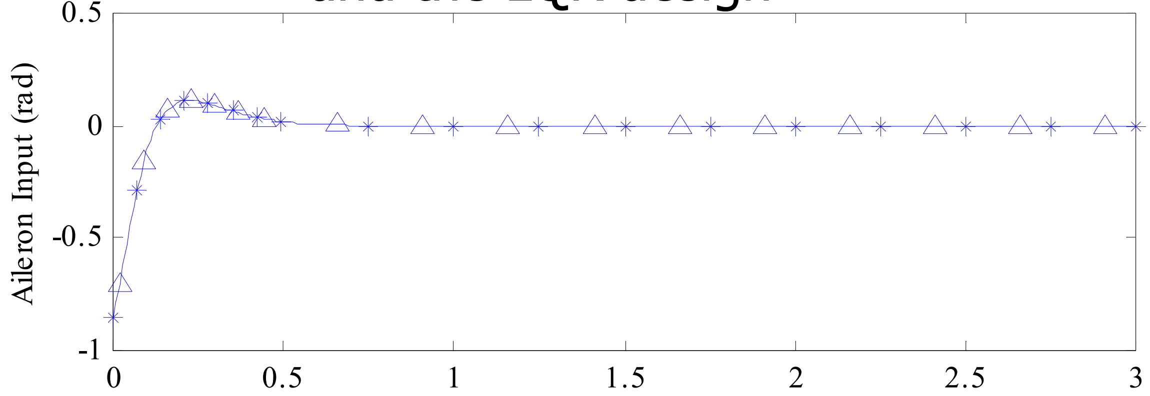


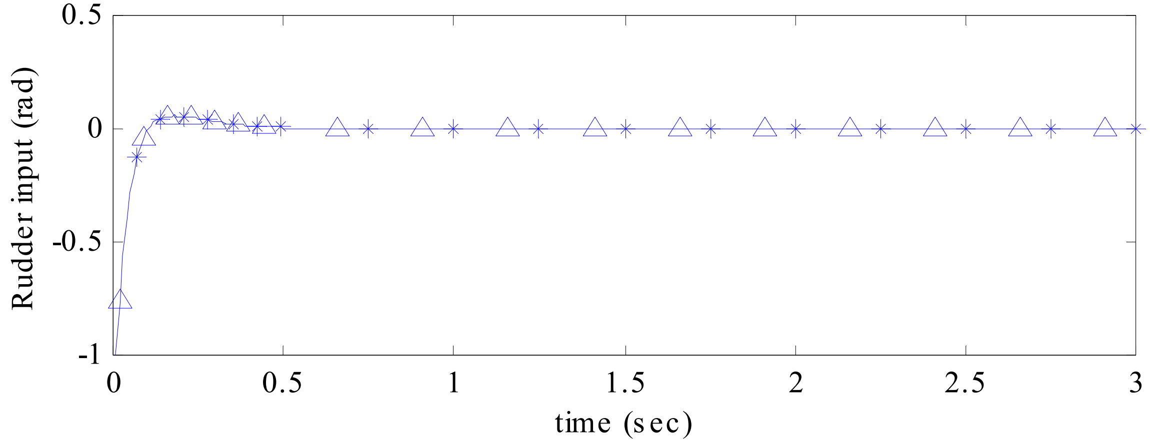
△△△ - State trajectories using the cost-to-go design (r = 35)

*** - State trajectories using the cost-to-go design (r = 50)

——— - State trajectories using LQR based optimal control

Comparison of the control trajectories using the cost-to-go design and the LQR design





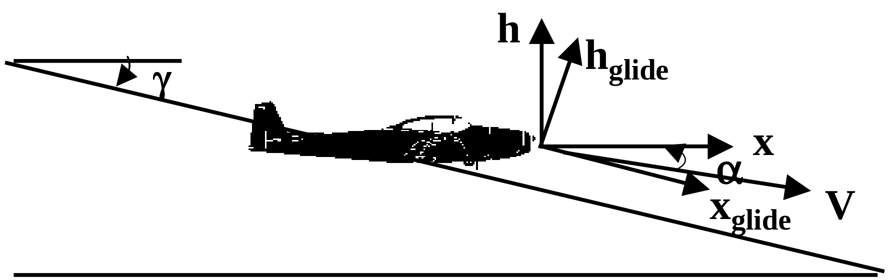
△△△ - Control trajectories using the cost-to-go design (r = 35)

*** - Control trajectories using the cost-to-go design (r = 50)

—— - Control trajectories using LQR based optimal control

Nonlinear Control of Aircraft in an approach configuration

Aircraft in an approach configuration



Nominal Flight Conditions

V_{nom} (ft/s)	γ_{nom} (deg)	α_{nom} (deg)	T_{nom} (lb)
235	-3	3.6	16800

Equations of motion in the wind-axes system

$$\dot{X} = V\cos\gamma$$

$$\dot{h} = V\sin\gamma$$

$$\dot{V} = \frac{T}{m}\cos\alpha - \frac{D}{m} - g\sin\gamma$$

$$\dot{\gamma} = \frac{T}{mV}\sin\alpha + \frac{L}{mV} - \frac{g}{V}\cos\gamma$$

$$\dot{T} = \frac{(\delta T - T)}{\tau_e}$$

Aircraft Parameters

M (slugs)	T_{max} (lb)	S_{ref} (ft^2)	C_{L0}	$C_{L\alpha}$	C_{D0}	τ_e	ε
4660	42000	1560	1.36	5.04	0.064	4	0.067

Implementation Details

- Equations are written with a change of coordinates while maintaining the nonlinearity.

$$\Delta \boldsymbol{x}(t) = \boldsymbol{x}(t) - \boldsymbol{x}_{nom}(t)$$
$$= \left[\boldsymbol{X}(t) - \boldsymbol{X}_{nom}(t) \;\; \boldsymbol{h}(t) - \boldsymbol{h}_{nom}(t) \;\; \boldsymbol{V}(t) - \boldsymbol{V}_{nom} \;\; \gamma(t) - \gamma_{nom} \;\; \boldsymbol{T}(t) - \boldsymbol{T}_{nom}\right]^{T}$$
$$\Delta \boldsymbol{u}(t) = \boldsymbol{u}(t) - \boldsymbol{u}_{nom}$$
$$= \left[\alpha(t) - \alpha_{nom} \;\; \delta\boldsymbol{T}(t) - \delta\boldsymbol{T}_{nom}\right]^{T}$$

- ΔX and Δh are transformed through a coordinate transformation,

$$\Delta \boldsymbol{X}_{approach} = \Delta \boldsymbol{h} \sin \gamma_{nom} + \Delta \boldsymbol{X} \cos \gamma_{nom}$$
$$\Delta \boldsymbol{h}_{approach} = \Delta \boldsymbol{h} \cos \gamma_{nom} - \Delta \boldsymbol{X} \sin \gamma_{nom}$$

so that now they represent perturbations along and perpendicular to the approach slope and we can now ignore the dynamics of the perturbation along the approach slope.

Implementation Details…

- Equations of motion in terms of the nonlinear perturbation dynamics:

$$\Delta \dot{h}_{approach} = (V_{nom} + \Delta V)\sin(\Delta\gamma)$$

$$\Delta \dot{V} = \frac{(T_{nom} + \Delta T)}{m}\cos(\alpha_{nom} + \Delta\alpha) - \frac{D}{m} - g\sin(\gamma_{nom} + \Delta\gamma)$$

$$\Delta \dot{\gamma} = \frac{(T_{nom} + \Delta T)}{m(V_{nom} + \Delta V)}\sin(\alpha_{nom} + \Delta\alpha) + \frac{L}{m(V_{nom} + \Delta V)} - \frac{g}{(V_{nom} + \Delta V)}\cos(\gamma_{nom} + \Delta\gamma)$$

$$\Delta \dot{T} = \frac{(\Delta\delta T - \Delta T)}{\tau_e}$$

$$\Delta x = \begin{bmatrix}\Delta h_{approach} & \Delta V & \Delta\gamma & \Delta T\end{bmatrix}^T ; \quad \Delta A = \begin{bmatrix}\Delta\alpha & \Delta\delta T\end{bmatrix}^T$$

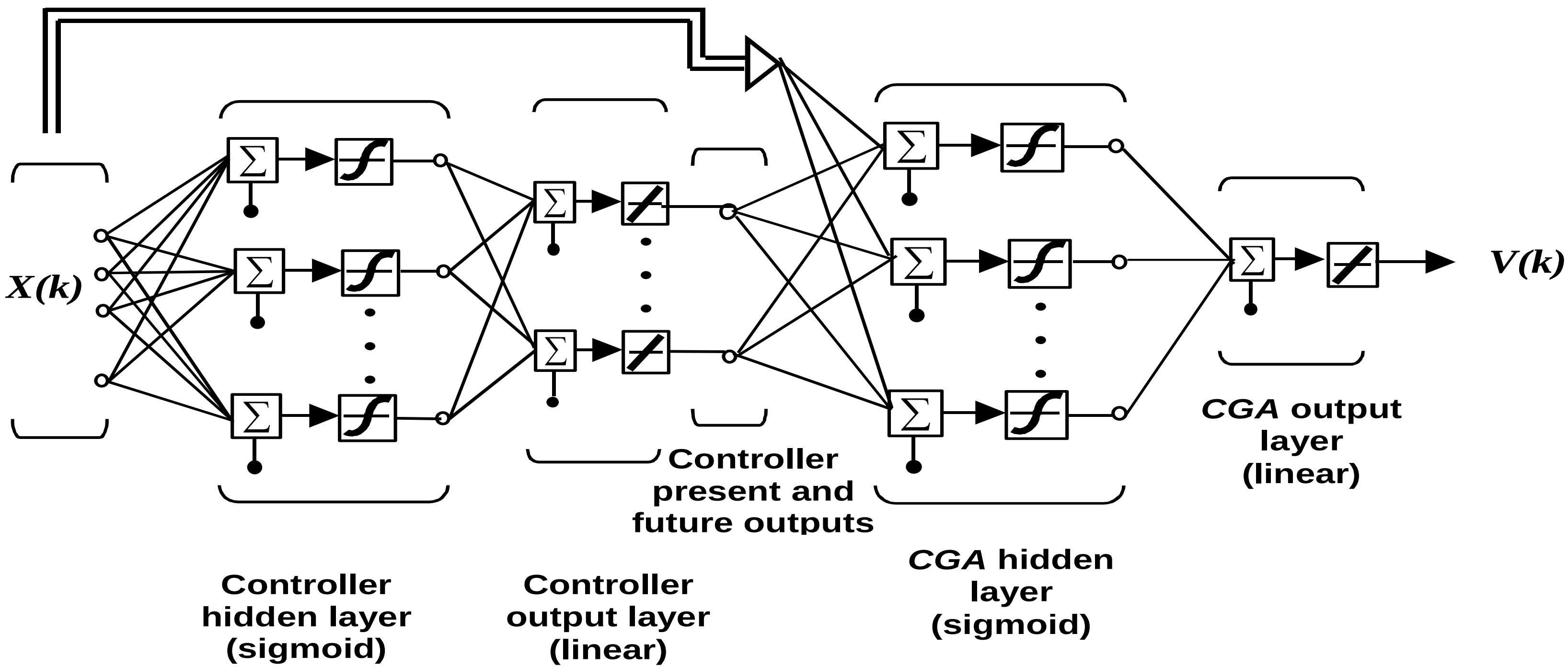
- Equations are discretized with a time step of 0.5 seconds.
- Specification of the cost function:

$$J = \frac{1}{2}\sum_{i=1}^{t_f/\Delta t}\left[\Delta x(i+1)^T Q \Delta x(i+1) + \Delta u(i)^T R \Delta u(i)\right]$$

$$Q = diag([10^{-4} \quad 10^{-2} \quad 1 \quad 10^{-8}])$$

$$R = diag([1 \quad 10^{-8}])$$

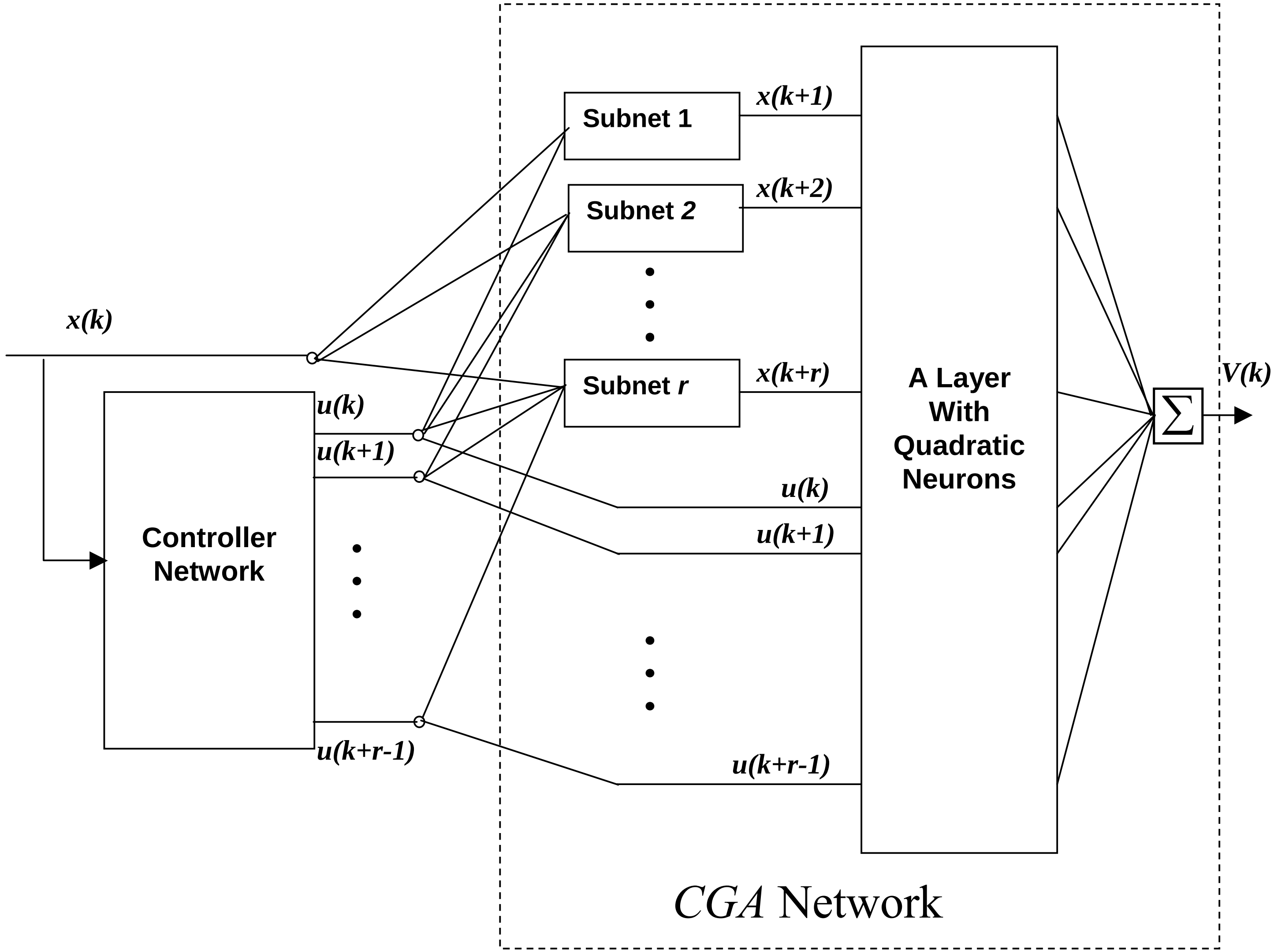
Control Architecture



Combined Neural Network having the CGA Network in front of the Controller Network.

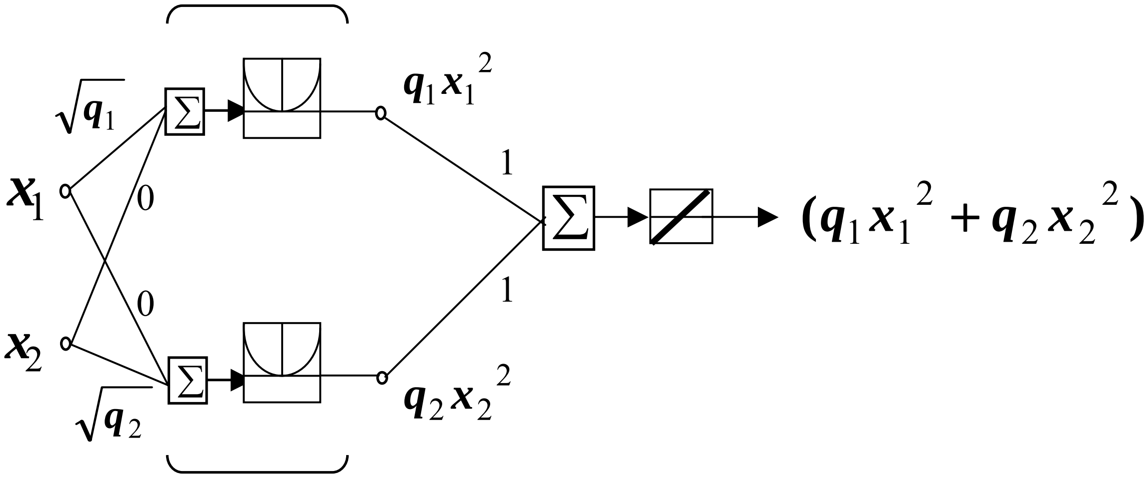
- Forming a combined Neural Network
- Fix the weights of the CGA part of the Network
- Training inputs to the network: Random values of *x(k)*
- Train the Network so that it gives the output value of zero for all the input random *x(k)*

Bringing Structure to the *CGA* Network



A Control Architecture Proposed to Simplify the Neural Network Training Problem

Implementation of the quadratic layer

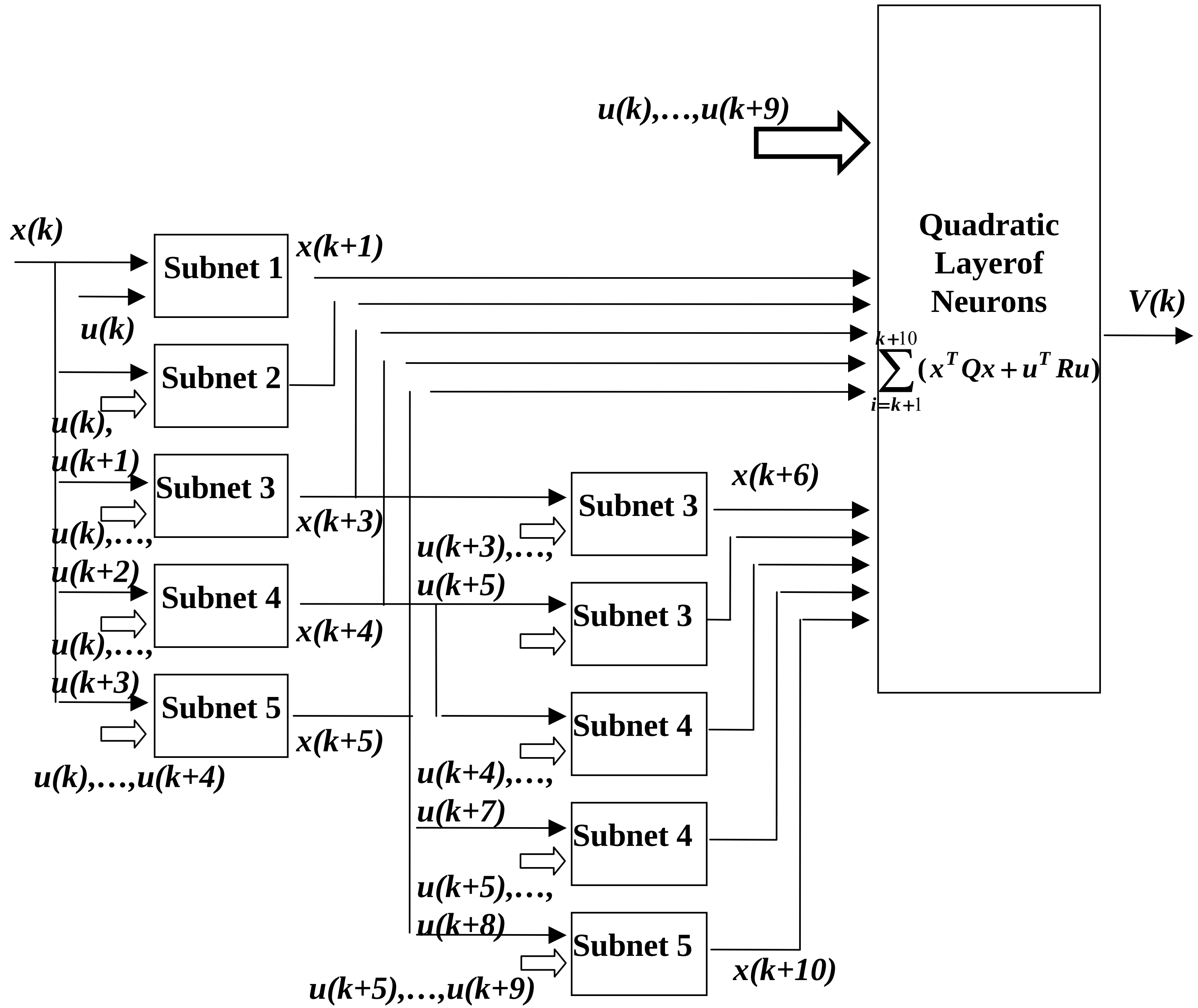


Layer with square neurons

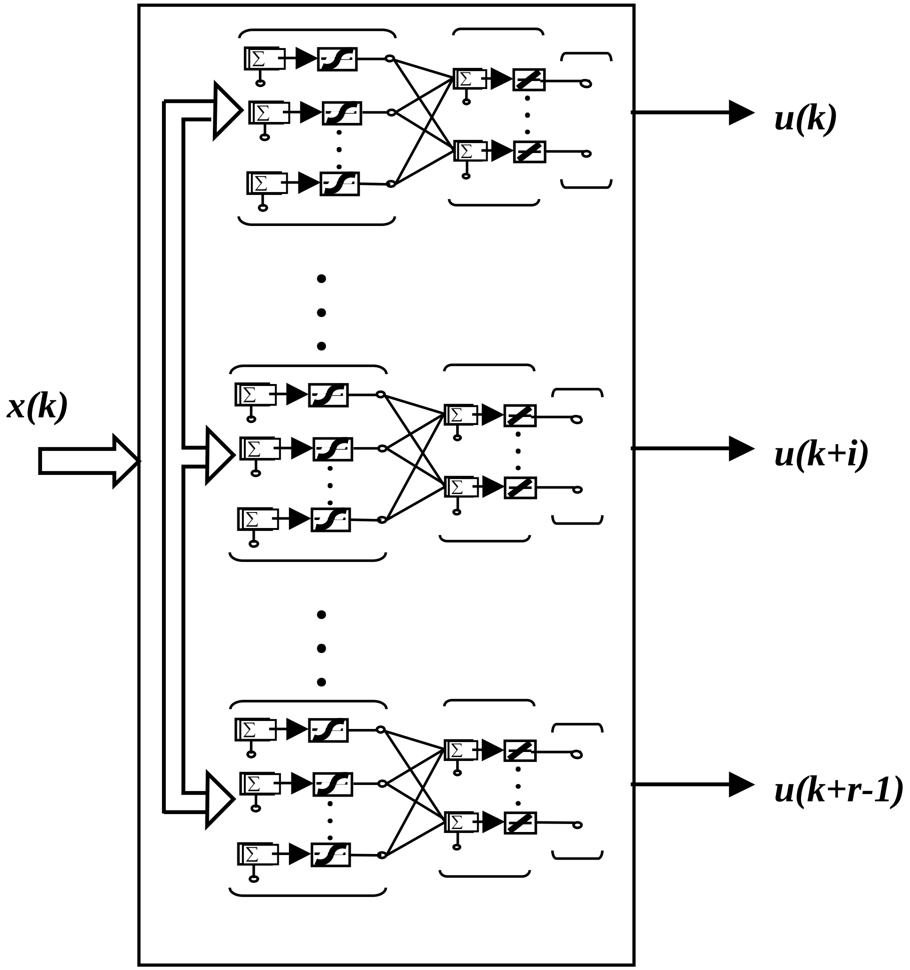
Advantages of the new structure:

- Guaranteed positive definiteness.
- Replaced training of a complex function by by the training of several simpler functions. functions.
- A good quality control ability.
- Allow for hybrid architecture

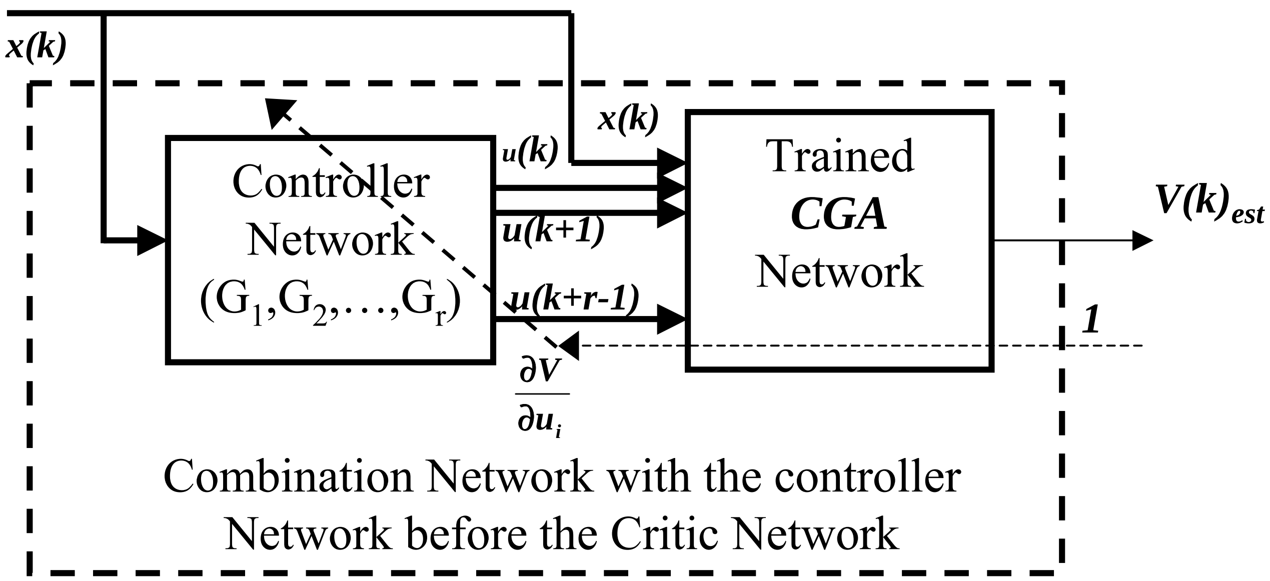
Implementation of the Hybrid CGA Network of order ‘r = 10’, using trained subnets of order 1 through 5



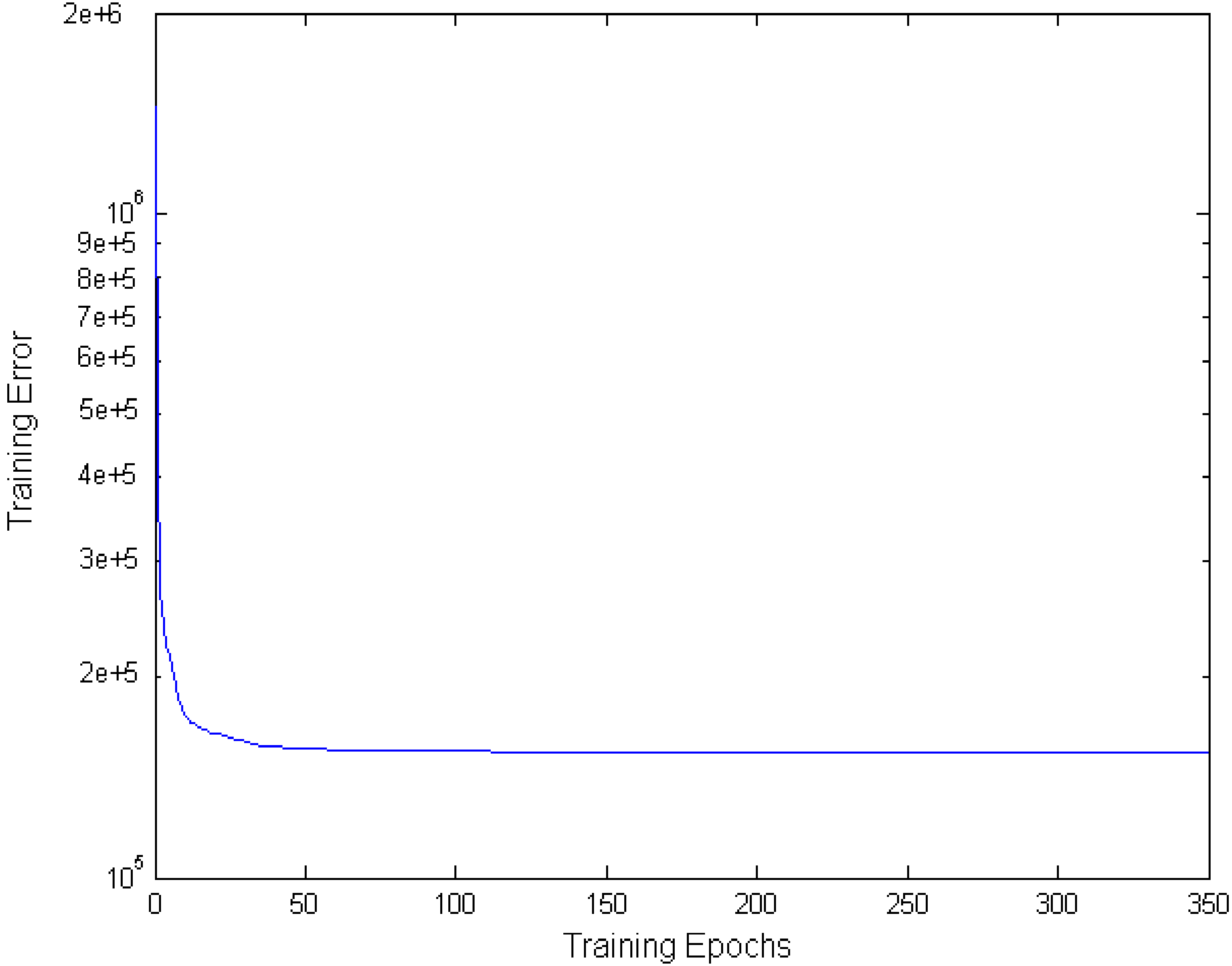
Internal Structure of the Neural Network Controller showing the separate Controller Subnets



Neural Network Controller training using the trained *CGA* Network

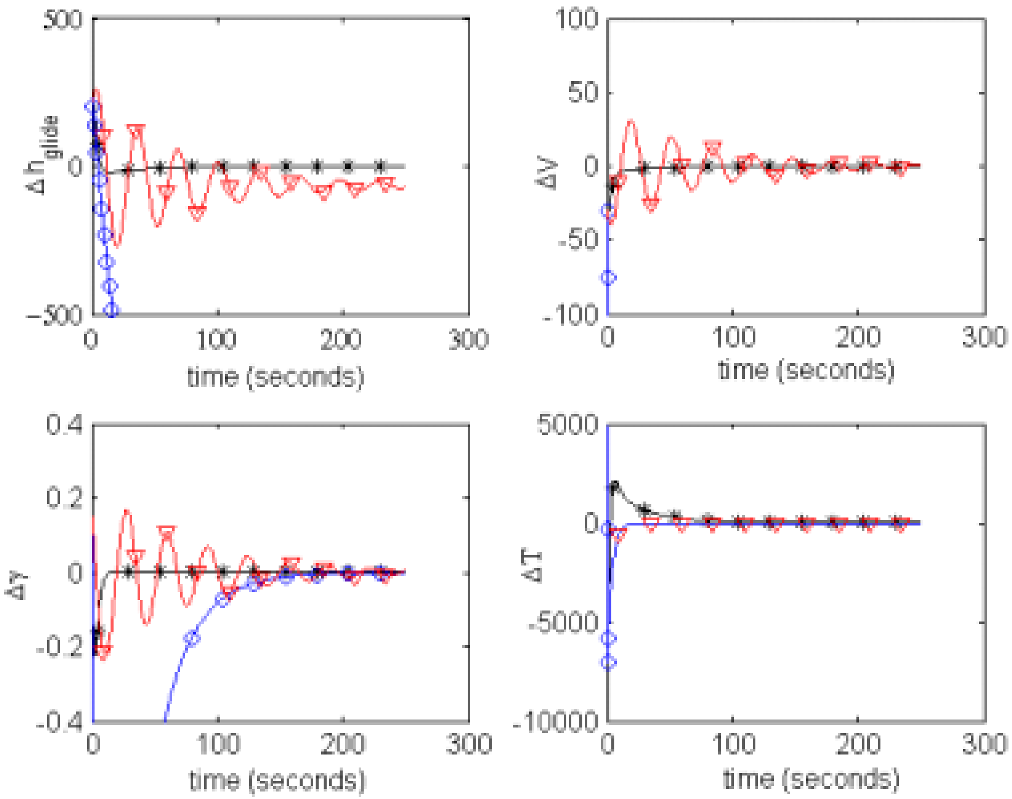


Cumulative value of V(k) getting minimized with training of the Neural Network Controller Weights



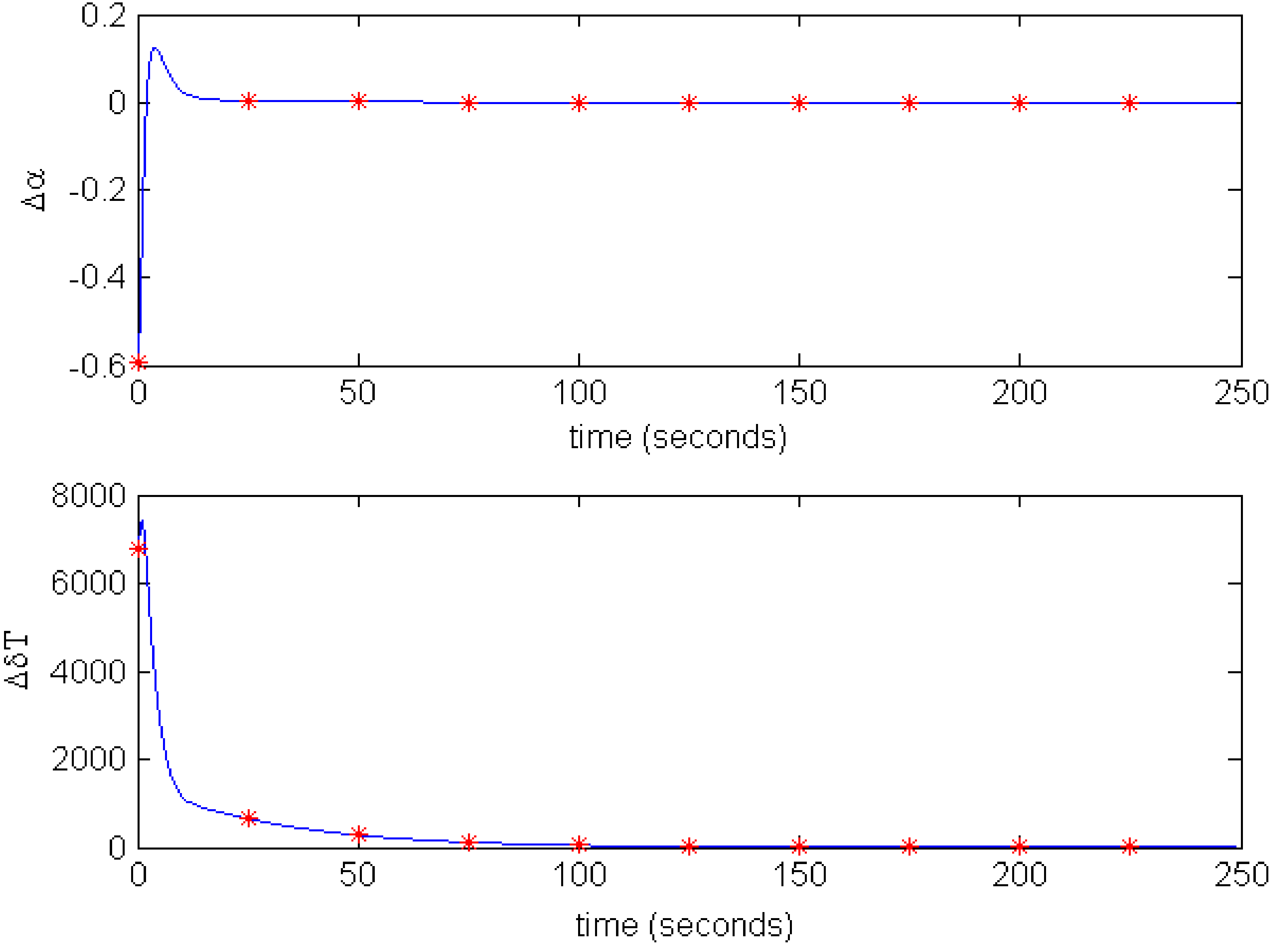
Aircraft response after an Initial perturbation with and without control

$$\Delta X(1) = [200 \quad -30 \quad 0.15 \quad -7000]^T$$



- ***∇-Open loop dynamics, $*$- Response with the Optimal Nonlinear Neural Network Controller, 'O' – Response with the LQR***

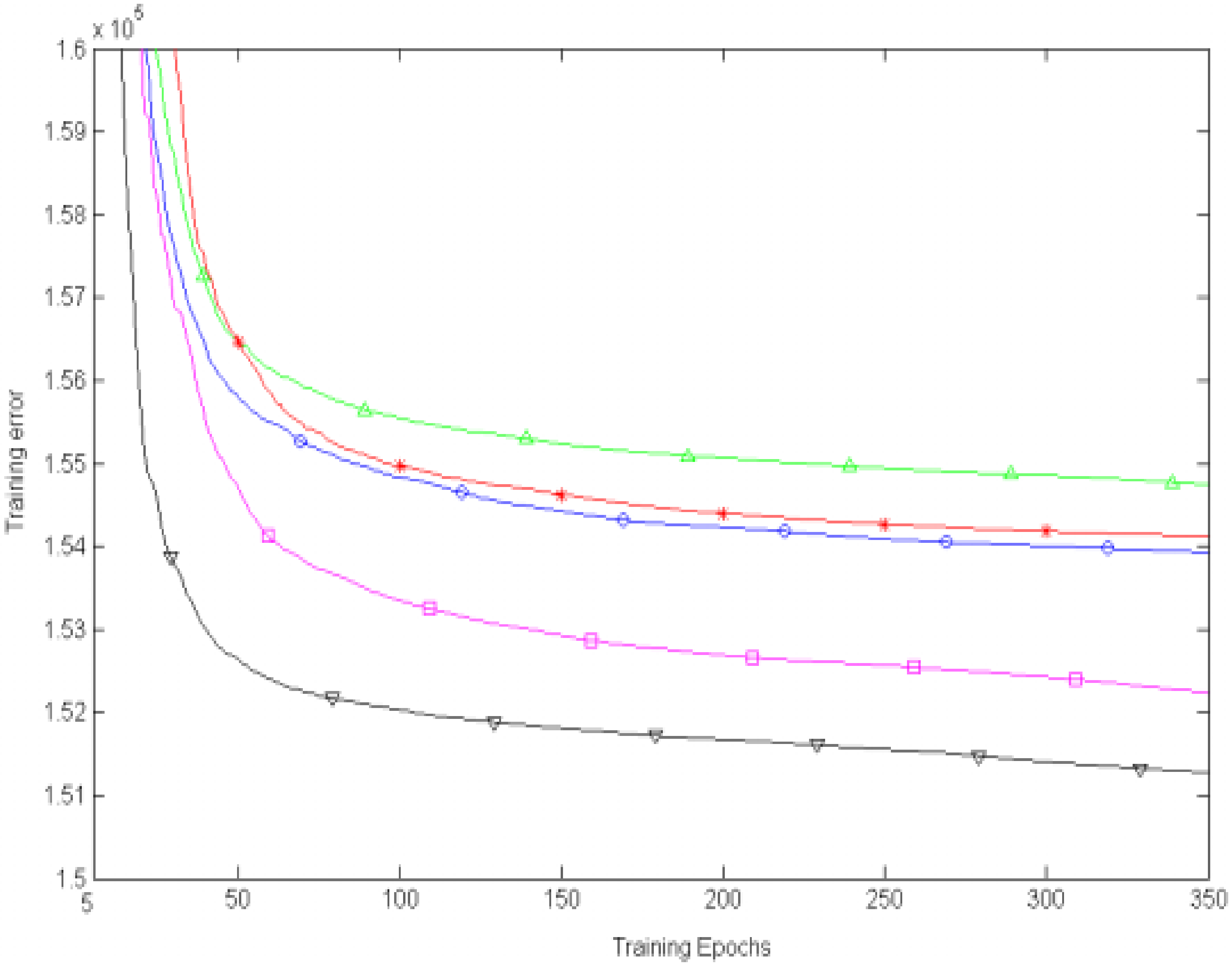
Neural Network Controller Outputs



A comparison of the cost function, J, as a function of the order of approximation, '*r*' of V(k)

'*r*'	'*J*'
5	410.8852
10	168.2999
15	93.6265
20	92.4814
25	88.5529

Nonlinear Optimization- Global or Local



Conclusions:

- New Neural Network Control Architecture for optimal control.
- Applicable to both linear and nonlinear systems
- Data based.
- Systematic training procedure.
- Confirmation on a Nonlinear Aircraft Model.